

Object Oriented concept in Ruby

Talk to a Teacher

<http://spoken-tutorial.org>

National Mission on Education through ICT

<http://sakshat.ac.in>

Anjana Nair

30 December 2013



Learning Objectives

In this tutorial we will learn to use



Learning Objectives

In this tutorial we will learn to use

- **classes**



Learning Objectives

In this tutorial we will learn to use

- classes
- **creating objects**



Learning Objectives

In this tutorial we will learn to use

- classes
- creating objects
- **different ways of defining methods in Ruby**



System Requirements

Here we are using



System Requirements

Here we are using

- **Ubuntu version 12.04**



System Requirements

Here we are using

- Ubuntu version 12.04
- Ruby 1.9.3



Prerequisites

To follow this tutorial, you must have



Prerequisites

To follow this tutorial, you must have

- **Internet Connection**



Prerequisites

To follow this tutorial, you must have

- Internet Connection
- Knowledge of Linux commands, Terminal and Text-editor



Prerequisites

To follow this tutorial, you must have

- Internet Connection
- Knowledge of Linux commands, Terminal and Text-editor
- If not, for relevant tutorials, please visit <http://spoken-tutorial.org>



Ruby's Object Oriented Nature

- Ruby is an object oriented language.



Ruby's Object Oriented Nature

- Ruby is an object oriented language.
- Everything in Ruby is an object; from a value to a string or number.



What is a Class in Ruby?

- **A class is a collection of related data and functions.**



What is a Class in Ruby?

- A **class** is a collection of related data and functions.
- It can serve to keep information organized.



What is a Class in Ruby?

- A **class** is a collection of related data and functions.
- It can serve to keep information organized.
- An **object** is an instantiation of a class .



Defining a class in Ruby

- A class definition begins with the keyword **class**.



Defining a class in Ruby

- A **class** definition begins with the keyword **class**.
- It is followed by the name of the **class**.



Defining a class in Ruby

- A **class** definition begins with the keyword **class**.
- It is followed by the name of the **class**.
- It is delimited with an end.



A class in Ruby

- Here is how you would write a **class** in Ruby:



A class in Ruby

- Here is how you would write a **class** in Ruby:



A class in Ruby

- Here is how you would write a **class** in Ruby:

```
class Product  
  ruby code  
end
```



A class in Ruby

- Here is how you would write a **class** in Ruby:

```
class Product  
  ruby code  
end
```

- The name of the **class** must begin with a capital letter.



Classes in Ruby contd...

- Names that contain more than one word should be camelcased



Classes in Ruby contd...

- Names that contain more than one word should be camelcased



Classes in Ruby contd...

- Names that contain more than one word should be camelcased
- **For example:**



Classes in Ruby contd...

- Names that contain more than one word should be camelcased
- For example:
- 1) **UserInformation**



Classes in Ruby contd...

- Names that contain more than one word should be camelcased
- For example:
 - 1) `UserInformation`
 - 2) `ProductInformation`



Classes in Ruby contd...

- Names that contain more than one word should be camelcased
- For example:
 - 1) UserInformation
 - 2) ProductInformation
- The subsequent file names will have underscore separating the words:



Classes in Ruby contd...

- Names that contain more than one word should be camelcased
- For example:
 - 1) `UserInformation`
 - 2) `ProductInformation`
- The subsequent file names will have underscore separating the words:
 - 1) `user_information.rb`



Classes in Ruby contd...

- Names that contain more than one word should be camelcased
- For example:
 - 1) `UserInformation`
 - 2) `ProductInformation`
- The subsequent file names will have underscore separating the words:
 - 1) `user_information.rb`
 - 2) `product_information.rb`



What is an object?

- **An object is an instance of a class.**



What is an object?

- An **object** is an **instance** of a **class**.



What is an object?

- An **object** is an **instance** of a **class**.
- Which means an **object** is created from a **class**.



What is an object?

- An **object** is an **instance** of a **class**.
- Which means an **object** is created from a **class**.



What is an object?

- An **object** is an **instance** of a **class**.
- Which means an **object** is created from a **class**.
- An object will have the properties and methods defined in the **class**.



What is an object?

- An **object** is an **instance** of a **class**.
- Which means an **object** is created from a **class**.
- An object will have the properties and methods defined in the **class**.



What is an object?

- An **object** is an **instance** of a **class**.
- Which means an **object** is created from a **class**.
- An object will have the properties and methods defined in the **class**.



How do you declare an object?

- We declare an object of a class using the new keyword.



How do you declare an object?

- We declare an object of a **class** using the **new** keyword.



How do you declare an object?

- We declare an object of a **class** using the **new** keyword.



Declaring an object in Ruby

- Here we are declaring an object of the Product class.



Declaring an object in Ruby

- Here we are declaring an **object** of the **Product class**.



Declaring an object in Ruby

- Here we are declaring an **object** of the **Product class**.
- **Here an object gets created.**



Declaring an object in Ruby

- Here we are declaring an **object** of the **Product class**.
- Here an **object** gets created.



Declaring an object in Ruby

- Here we are declaring an **object** of the **Product class**.
- Here an **object** gets created.
- **product = Product.new**



Declaring an object in Ruby

- Here we are declaring an **object** of the **Product class**.
- Here an **object** gets created.
- `product = Product.new`



Declaring an object in Ruby

- Here we are declaring an **object** of the **Product class**.
- Here an **object** gets created.
- `product = Product.new`
- **This process is called initialization of an object.**



Declaring an object in Ruby

- Here we are declaring an **object** of the **Product class**.
- Here an **object** gets created.
- `product = Product.new`
- This process is called initialization of an **object**.
- **This object is of type : Product.**



Objects and the initialize method

- Now let us look at what an initialize method is.



Objects and the initialize method

- Now let us look at what an initialize method is.



Objects and the initialize method

- Now let us look at what an initialize method is.
- An initialize method is called at the time of object creation.



Objects and the initialize method

- Now let us look at what an initialize method is.
- An initialize method is called at the time of **object** creation.



Objects and the initialize method

- Now let us look at what an initialize method is.
- An initialize method is called at the time of **object** creation.
- On calling "new" on an **object** we invoke the initialize method.



The initialize method

- An initialize method may take a list of parameters.



The initialize method

- An **initialize** method may take a list of parameters.



The initialize method

- An **initialize** method may take a list of parameters.
- Like other Ruby methods, it is preceded by the **def** keyword.



The initialize method

- An **initialize** method may take a list of parameters.
- Like other Ruby methods, it is preceded by the **def** keyword.



The initialize method

- An **initialize** method may take a list of parameters.
- Like other Ruby methods, it is preceded by the **def** keyword.
- **Let us look at an example.**



Defining methods in a class

- Recall that in Ruby, methods are the functions that a **class** performs.



Defining methods in a class

- Recall that in Ruby, methods are the functions that a **class** performs.



Defining methods in a class

- Recall that in Ruby, methods are the functions that a **class** performs.
- **A multiword method name is separated with an underscore.**



Defining methods in a class

- Recall that in Ruby, methods are the functions that a **class** performs.
- A multiword method name is separated with an underscore.



Defining methods in a class

- Recall that in Ruby, methods are the functions that a **class** performs.
- A multiword method name is separated with an underscore.



Defining methods in a class

contd...

- Some of the characters that can be appended to a method name are:



Defining methods in a class

contd...

- Some of the characters that can be appended to a method name are:



Defining methods in a class

contd...

- Some of the characters that can be appended to a method name are:
- ?



Defining methods in a class

contd...

- Some of the characters that can be appended to a method name are:
- ?
- =



Defining methods in a class

contd...

- Some of the characters that can be appended to a method name are:
- ?
- =
- Each of the characters add some meaning to the method.



Defining methods in a class

contd...

- Some of the characters that can be appended to a method name are:
- ?
- =
- Each of the characters add some meaning to the method.
- Let us look at some examples.



Summary

In this tutorial we have learnt:

- How to declare **classes**
- How to create **objects** of a **class**
- Different ways of defining methods in Ruby



Assignment

As an assignment

- Define a class **Product**



Assignment

As an assignment

- Define a class **Product**
- Define methods that you can use get values of myvar and set values for myvar.



Assignment

As an assignment

- Define a class **Product**
- Define methods that you can use get values of myvar and set values for myvar.
- To set values define the method using **= sign.**



Assignment

As an assignment

- Define a class **Product**
- Define methods that you can use get values of myvar and set values for myvar.
- To set values define the method using = sign.
- **Instantiate the object of the class and set and get values using the above 2 methods.**



About the Spoken Tutorial Project

- Watch the video available at http://spoken-tutorial.org/What_is_a_Spoken_Tutorial
- It summarises the Spoken Tutorial project



About the Spoken Tutorial Project

- Watch the video available at http://spoken-tutorial.org/What_is_a_Spoken_Tutorial
- It summarises the Spoken Tutorial project
- If you do not have good bandwidth, you can download and watch it



Spoken Tutorial Workshops

The Spoken Tutorial Project Team

- Conducts workshops using spoken tutorials
- Gives certificates to those who pass an online test
- For more details, please write to contact@spoken-tutorial.org



Acknowledgements

- Spoken Tutorial Project is a part of the Talk to a Teacher project
- It is supported by the National Mission on Education through ICT, MHRD, Government of India
- More information on this Mission is available at <http://spoken-tutorial.org/NMEICT-Intro>

